Rapport du Projet GLCS

KHAROUNI Yanis, LIU Zhiyuan

28 Janvier 2021

1 Fonctionnement du programme

1.1 Class *CartesianDistribution2D*

Ce class représent un élément basic donc un processus, de la structure de donnée principale. Il contient un constant, sa dimension égale à 2, et un communicateur MPI. Il contiens plusieurs méthodes internes comme suit :
CartesianDistribution2D : Créer et initialiser ce class sous le communicateur et la dimension donnés.
extents : Obtenir la forme de la distribution des processus sur la grille cartésienne.
coord : Obtenir les cordonnées du processus actuel.
neighbour_rank : Obtenir le rang d'un processus voisin selon la direction donnée en paramètre

1.2 Fonction *Distributed2DField::sync_ghosts*

Cette fonction réalise la synchronisation des zones fantômes pendant le calcul parallèle via MPI.

2 Ajout de fonctionnalités

2.1 Input des paramètres

Dans le programme original, les paramètres du programme devront être entrée manuellement par ligne de commande chaque fois que l'on lance le programme. On vise d'implémenter une méthode d'input lisible, compréhensible

et bien structurée. On a choisi d'utiliser la bibliothèque *Boost.Program_options*. Cette bibliothèque nous permet de gérér l'input de paramètres de façon systématique.

2.1.1 Implémentation

Cette fonction est réalisée par l'implémentation d'une bibliothèque indépendante *programoptions*. Comme les autres bibliothèques, elle est composée de son header, fiche .c et un CMakeLists. Son répertoire est ajouté puis relié par le CMakeLists principal en utilisant *add_subdirectory* et *target_link_libraries*.

Le but initial était de lire des fichiers de configurations qui contiennent les paramètres détaillés. Le non du fichier choisi serait entré par ligne de commande comme un paramètre, avec une valeur par défaut *config.conf*. Au même temps on gardrait la possibilité d'entrer tous les paramètres en ligne de commande en ajoutant leurs préfixes comme *-N 10 -d 0.125* et etc.

Néanmoins on n'a pas pu réaliser cette fonctionnalité à cause de l'erreur *terminate called after throwing an instance of 'boost::....* On n'a pas pu trouver une solution. Les codes inutiles sont mis à coté et annotés.

2.1.2 Résultats

La dernière version de cette bibliothèque permet de recevoir les paramètres seulement par un fichier de configuration, organisé de façon suivante :
Le nouveau class *BoostConfig* équivalent au *CommandLineConfig* cotient

```
max_iteration=10
global_shape=4
global_shape=8
dist_extents=1
dist_extents=2
delta_t=0.125
delta_s=1
delta_s=1
```

aussi les méthodes qui renvoient ses variables internes.

2.2 Stokage de donnees de sortie dans un fichier HDF5

Le programme initial affiche directement sur console les resultats de la simulation et ceci ne permet pas a l'utilisateur de garder les donnees apres avoir termine la simulation c'est pourquoi on a propose une classe permettant a l'utilisateur de stocker le resultat dan un fichier de format HDF5 en utilisant la bibliotheque hdf5.

2.3 Implementation

Cette fonction est réalisée par l'implémentation d'une bibliothèque indépendante *hdf5data*. Comme les autres bibliothèques, elle est composée de son header, fiche .c et un CMakeLists. Son répertoire est ajouté puis relié par le CMakeLists principal en utilisant *add_subdirectory* et *target_link_libraries*.

3 Compilation

3.1 Utilisation de HDF5

En ce qui concerne la compilation du projet avec hdf5, afin de pouvoir utiliser la bibliotheque hdf5 on a du ajouter l'instructions suivante au CMakeLitsts du porjet: *find_package(HDF5 REQUIRED COMPONENTS C HL)* pour trouver le package de la bibliotheque HDF5 puis *hdf5::hdf5* a l'instruction *target_link_librairies* afin de lier l'exetuable a la bibliotheque HDF5

3.2 Utilisation de Boost

Egalement pour la compilation du projet avec la biblotheque boost, l'instructions suivante a ete ajoutee au CMakeLitsts du porjet: *FIND_PACKAGE(Boost COMPONENTS program_options REQUIRED)* pour trouver le package de la bibliotheque boost puis *{$Boost_LIBRARIES}* a l'instruction *target_link_librairies* pour lier l'exetuable a la bibliotheque boost